Computers and Mathematics with Applications 60 (2010) 2279–2285

Contents lists available at ScienceDirect

Computers and Mathematics with Applications

journal homepage: www.elsevier.com/locate/camwa

An electromagnetism-like method for nonlinearly constrained global optimization

M.M. Ali [a,*], M. Golalikhani [b]

[a] School of Computational and Applied Mathematics, Witwatersrand University, Wits 2050, Johannesburg, South Africa

[b] Department of Industrial and Systems Engineering, University at Buffalo, The State University of New York, Buffalo, NY 14260, United States

ARTICLE INFO

Article history:
Received 14 February 2010
Received in revised form 6 August 2010
Accepted 6 August 2010

Keywords:
Constrained global optimization
Electromagnetism-like method
Attraction–repulsion mechanism
Meta-heuristics

ABSTRACT

We propose an electromagnetism-like (EM) method for constrained global optimization. The method is a modified version of the unconstrained EM method. We introduce the charge calculation of a point based on both the function value and the total constraint violations. Hence, the calculation of the total force vector is different from the original EM method. The new method is not penalty function-based and therefore the difficulty with the choice of the penalty parameter value does not arise. We have tested our method on a set of 13 benchmark test problems. Results obtained are compared with those from some recent algorithms. The comparisons show that our proposed method is suitable for solving constrained optimization problems.

1. Introduction

Constrained global optimization problems arise in many applications in science and engineering. These problems can be mathematically formulated as the following: given a real objective function f defined on a feasible set $\Omega \subset \mathbb{R}^n$, find a point $x^* \in \Omega$ and the corresponding value f^* such that

$$f^* = f(x^*) = \min \{f(x) \mid \forall x \in \Omega\}. \tag{1}$$

We assume that the feasible set Ω is non-empty and its measure is positive, i.e. $m(\Omega) > 0$. The set Ω is bounded by boundary constraints, $X = [l, u]$, and m inequality constraints, i.e.

$$\Omega = \left\{x \in \mathbb{R}^n \mid x \in X \text{ and } g_i(x) \leq 0 \text{ for } i = 1, 2, \ldots, m\right\}, \tag{2}$$

where $l, u \in \mathbb{R}^n$ and $g_i : \mathbb{R}^n \to \mathbb{R}$. We denote Eqs. (1)–(2) as problem (P). Formally, the problem (P) is defined as

$$(P) \quad \begin{cases} \min & f(x), \\ \text{such that} & g_i(x) \leq 0, \quad i = 1, 2, \ldots, m, \\ & x \in X. \end{cases}$$

In constrained optimization problems like problem (P), optimal solutions usually lie on the boundary of the feasible region. Hence, in the context of constrained global optimization, the problem (P) is most often converted into an unconstrained problem via a penalty function. This helps with the exploration of both feasible and infeasible regions. In the penalty function approach, a penalty term is added to the objective function to penalize the constraint violation. An advantage of this approach is that the existing implementation of some unconstrained (or bound constrained) global optimization algorithms

* Corresponding author. Tel.: +27 117176139; fax: +27 117176149.
E-mail address: Montaz.Ali@wits.ac.za (M.M. Ali).

0898-1221/$ – see front matter © 2010 Elsevier Ltd. All rights reserved.
doi:10.1016/j.camwa.2010.08.018

can be easily used by simply modifying the objective function. As a result, a number of genetic/evolutionary algorithms for constrained global optimization have been suggested in the literature [1,2], where their existing implementations have been used.

In the penalty function-based methods, the inherent difficulty of the choice of penalty parameter is well documented [2]. Runarsson and Yao [2] discussed the effects of underpenalization (which may produce a solution outside Ω) and overpenalization (an optimal solution in $\partial\Omega$ may never be found), and suggested the so-called stochastic ranking procedure to overcome the difficulties with them. However, the stochastic ranking is not parameter-free. Indeed, the penalty parameter is implicitly represented by a user provided probability in the stochastic ranking procedure.

Other researchers modified the penalty function to make the penalty parameter less sensitive; see for example the superiority of feasible points (SFP) [3] and the parameter-free penalty (PFP) [4]. However, it can be shown that these methods cause overpenalization [2] since the infeasible solutions are always regarded as worse than the feasible ones. Clearly, the penalty parameter cannot be made insensitive for any arbitrary problem, as this will depend on whether the constraint violation or the function value is dominant. A better approach is therefore to design a method which uses the constraint violation but where the penalty parameter is not needed. In this paper, we try to achieve this by designing an EM method for constrained global optimization.

The EM method was first introduced by Birbil and Fang [5]. It simulates the attraction–repulsion mechanism in the electromagnetism theory [6] in order to solve unconstrained or bound constrained global optimization problems. Despite its proven capability for solving various unconstrained optimization problems [5,7–9], no credible paper on the EM method exists for solving constrained optimization problems except the conference paper of Rocha and Fernandes [10]. The constrained electromagnetism-like (CEM) method proposed in this paper is completely different from the method suggested in [10].

The next section begins by briefly introducing the EM method and this is followed by the CEM method, presented in Section 3. In Section 4, we present a full set of numerical results and compare the CEM method with a number of recent algorithms. Concluding remarks are made in Section 5.

2. The electromagnetism-like method

Initially designed for bound constrained optimization problems, the EM method [5] utilizes N, n-dimensional points $x_{i,k}$, $i = 1, 2, \ldots, N$, as a population for searching the feasible set

$$X = \left\{x \in R^n \mid l_i \leq x \leq u_i, i = 1, 2, \ldots, n\right\}.$$

The index k denotes the iteration (or generation) number of the algorithm. The initial population,

$$S_k = \left\{x_{1,k}, x_{2,k}, \ldots, x_{N,k}\right\}, \tag{3}$$

where $k = 1$, is taken to be uniformly distributed in the search region, X. We denote the population set at the k-th iteration by S_k, as the members of the set S_k change with k. After the initialization of S_k, EM continues its iterative process until a stopping condition (e.g. the maximum number of iterations) is met. An iteration of EM consists of two steps. In the first step, each point in S_k moves to a different location by using the attraction–repulsion mechanism of the electromagnetism theory [6]. In the second step, points moved by the electromagnetism theory are further moved locally by a local search and then become the members of S_{k+1} in the $(k+1)$-th iteration. Both the attraction–repulsion mechanism and the local search in EM are responsible for driving the members, $x_{i,k}$, of S_k to the close proximity of the global minimizer.

As with the electromagnetism theory for charged particles, each point $x_{i,k} \in S_k$ in the search space X is assumed as a charged particle where the charge of a point relates to its objective function value. Points with better objective function value have more charges than other points, and the attraction–repulsion mechanism is a process in EM by which points with more charge attract other points in S_k, and points with less charge repel other points. Finally, a total force vector, F_i^k, exerted on a point, e.g. the i-th point $x_{i,k}$, is calculated by adding these attraction–repulsion forces and each $x_{i,k} \in S_k$ is moved in the direction of its total force to the location $y_{i,k}$. A local search is used to explore the vicinity of each $y_{i,k}$ by shifting $y_{i,k}$ to $z_{i,k}$. The members, $x_{i,k+1} \in S_{k+1}$, of the $(k+1)$-th iteration are then found by using

$$x_{i,k+1} = \begin{cases} y_{i,k} & \text{if } f(y_{i,k}) \leq f(z_{i,k}) \\ z_{i,k} & \text{otherwise.} \end{cases} \tag{4}$$

Algorithm 1 shows the general scheme of EM. We also provide the description of each step following the algorithm.

Algorithm 1. [EM $(N, I_{\max}, I_\ell, \delta)$]

1. Input parameters: Input the maximum number of iterations $I_{\max}$, the values for the local search parameters such as I_ℓ and δ, and the size N of the population.
2. Initialize: Set the iteration counter $k = 1$, initialize the members of S_k uniformly in X, and identify the best point in S_k.
3. while $k < I_{\max}$ do
4. $\quad F_i^k \leftarrow \text{CalcF}(S_k)$

5. Move($x_{i,k}, F_i^k$)
6. Local($I_\ell, \delta, y_{i,k}$)
7. Select($S_{k+1}, y_{i,k}, z_{i,k}$)
8. $k = k + 1$
9. end while

Input parameter values (Line 1): the EM algorithm is run for $I_{\max}$ iterations. In the local search phase, $n \times I_\ell$ is the maximum number of locations $z_{i,k}$, within a δ distance of $y_{i,k}$, for each i.

Initialize (Line 2): The points $x_{i,k}$, $k = 1$, are selected uniformly in X, i.e. $x_{i,1} \sim Unif(X)$, $i = 1, 2, \ldots, N$, where *Unif* represents the uniform distribution. The objective function values $f(x_{i,k})$, $i = 1, 2, \ldots, N$, are computed, and the best point

$$x_k^b = Arg \min_{x_{i,k} \in S_k} \left\{ f(x_{i,k}) \right\} \tag{5}$$

is identified.

Calculate force (Line 4): In this step, a charged-like value ($q_{i,k}$) is assigned to each point ($x_{i,k}$). The charge $q_{i,k}$ of $x_{i,k}$ is dependent on $f(x_{i,k})$, and points with better objective function have more charge than others. The charges are computed as follows:

$$q_{i,k} = \exp\left(-n \frac{f(x_{i,k}) - f(x_k^b)}{\sum_{j=1}^{N} \left(f(x_{j,k}) - f(x_k^b) \right)} \right). \tag{6}$$

Then, the force, $F_{i,j}^k$, between two points, $x_{i,k}$ and $x_{j,k}$, is calculated by using

$$F_{i,j}^k = \begin{cases} (x_{j,k} - x_{i,k}) \dfrac{q_{i,k} q_{j,k}}{\|x_{j,k} - x_{i,k}\|^2} & \text{if } f(x_{i,k}) > f(x_{j,k}), \\ (x_{i,k} - x_{j,k}) \dfrac{q_{i,k} q_{j,k}}{\|x_{j,k} - x_{i,k}\|^2} & \text{if } f(x_{i,k}) \le f(x_{j,k}). \end{cases} \tag{7}$$

The total force, F_i^k, corresponding to $x_{i,k}$ is now calculated as

$$F_i^k = \sum_{j=1, j \neq i}^{N} F_{i,j}^k. \tag{8}$$

Move point $x_{i,k}$ along F_i^k (Line 5): In this step, each point $x_{i,k}$, except for x_k^b, is moved along the total force vector F_i^k using

$$x_{i,k} = x_{i,k} + \lambda \frac{F_i^k}{\|F_i^k\|} (RNG), \quad i = 1, 2, \ldots, N; i \neq b, \tag{9}$$

where $\lambda \sim Unif(0, 1)$ for each coordinate of $x_{i,k}$, and *RNG* denotes the allowed range of movement toward the lower or upper bound for the corresponding dimension.

Local search (Line 6): For each $y_{i,k}$ a maximum of I_ℓ points are generated in each coordinate direction in the δ neighborhood of $y_{i,k}$. This means that the process of generating local points is continued for each $y_{i,k}$ until either a better $z_{i,k}$ is found or the $n \times I_\ell$ trial is reached.[1]

Selection for the next iteration (Line 7): In this step, members $x_{i,k+1} \in S_{k+1}$ are selected from $y_{i,k}$ and $z_{i,k}$ using (4), and the best point is identified using (5).

3. The constrained EM (CEM) method

We now describe the CEM method for solving problem (P). There is no need to present a step by step algorithm for CEM as the basic structure of CEM is the same as that of EM. CEM is also population-based and it is implemented using all steps in Algorithm 1. The changes made lie in the concept of charge corresponding to each point $x_{i,k} \in S_k$, the calculation of the total force F_i^k, $i = 1, 2, \ldots, N$, and the selection (4) in Line 7 of Algorithm 1. To explain these changes, we begin by redefining the problem (P) in a multi-objective fashion as follows:

$$\min (f(x), \phi(x)), \quad x \in X, \tag{10}$$

[1] The local search used in the original EM is a very simple and naive coordinate direction search.

where $\phi(x)$ is the aggregate constraint violation at x, i.e.

$$\phi(x) = \sum_{j=1}^{m} \max\left[0, g_j(x)\right]. \tag{11}$$

Our intention however is not to use any multi-objective optimization in solving (P); we use Eqs. (10)–(11) for ease of explanation and presentation. We begin with the force calculation in CEM.

Calculate force (Line 4, Algorithm 1): The charge of $x_{i,k}$ now comprises two components, i.e. $(q^f_{i,k}, q^\phi_{i,k})$. The first component, $q^f_{i,k}$, of the charge is due to the objective function while the second component, $q^\phi_{i,k}$, is due to the aggregate constraint violation, $\phi(x_{i,k})$. The first component of the charge vector is calculated via (6). The second component is calculated using the same formula but by replacing $f(x)$ with $\phi(x)$, i.e.

$$q^\phi_{i,k} = \exp\left[-n\frac{\phi(x_{i,k}) - \phi(x^{lc}_{i,k})}{\sum_{j=1}^{N}\left(\phi(x_{i,k}) - \phi(x^{lc}_{i,k})\right)}\right], \tag{12}$$

where

$$x^{lc}_k = Arg\min_{x_{i,k}\in S_k}\left\{\phi(x_{i,k})\right\} \tag{13}$$

i.e., $x^{lc}_k \in S_k$ has the least constraint violation. The total charge corresponding to each point $x_{i,k} \in S_k$ is now defined as follows:

$$Q_{i,k} = \beta q^\phi_{i,k} + (1-\beta)q^f_{i,k}, \tag{14}$$

where $\beta \in [0, 1]$ is a parameter. For instance, if one is interested in getting to the feasible solution(s) quickly then β should be assigned a higher value. On the other hand, a smaller β would concentrate on minimization of $f(x)$, which may be useful for problems with large feasible region. A more cautious approach would be to choose β close to the midpoint 0.5, which we have done here, and this seems appropriate for most of the problems that we considered. We now define the force between $x_{i,k}$ and $x_{j,k}$, as in (7), by

$$F^k_{i,j} = \begin{cases} (x_{j,k} - x_{i,k})\dfrac{Q_{i,k}Q_{j,k}}{\|x_{j,k} - x_{i,k}\|^2} & \text{if } Q_{j,k} > Q_{i,k}, \\ (x_{i,k} - x_{j,k})\dfrac{Q_{i,k}Q_{j,k}}{\|x_{j,k} - x_{i,k}\|^2} & \text{if } Q_{j,k} \le Q_{i,k}. \end{cases} \tag{15}$$

The total force F^k_i corresponding to $x_{i,k} \in S_k$ and the movement of $x_{i,k}$ along F^k_i are similar to (8) and (9) respectively.

Selection of points in S_{k+1} (Line 7, Algorithm 1): Unlike EM, which uses (4), CEM selects the i-th point of S_{k+1} from $\{y_{i,k}, z_{i,k}\}$ using the following rules [4]:

- If one of the points $\{y_{i,k}, z_{i,k}\}$ is infeasible, we select the feasible one.
- If both points are infeasible,[2] we select the point which has the least $\phi(x)$.
- If both points are feasible, we select the point which has the least $f(x)$.

4. Numerical results and comparisons

In this section, we present the numerical results obtained from CEM and compare them with ones from two constrained versions of EM. These are just EM algorithms developed for solving constrained problems. The first version incorporates penalty functions in the original EM for solving constrained problems; see Birbil [11]. We denote this version by PEM. The second version was proposed by Rocha and Fernandes [10]. They used a set of feasibility and domain (FD) rules for handling constraints. We denote this version by FDEM. Finally, we compare CEM with four versions of particle swarm algorithms. These are original particle swarm optimization (PSO) [12], the particle evolutionary swarm optimization (PESO) [12], the constrained handling method of particle swarm optimization (CHMPSO) [13] and self-adaptive velocity particle swarm optimization (SAVPSO) [14]. For our numerical testing, we use a set of 13 benchmark test problems with dimensions ranging from 2 to 20. Full details of the problem set are given in [2]. There are four parameters in CEM and the values for these

[2] For the case where infeasible points cannot be evaluated, the following strategies can be adopted. When S_k contains at least one feasible point, a point, say x, from $\{y_{i,k}, z_{i,k}\}$ is selected at random and $f(x)$ and $\phi(x)$ are assigned in $[f(x_f), 5|f(x_f)|]$ at random where $f(x_f)$ is the function value at the worst feasible point in S_k. If however, S_k does not contain any feasible point then other heuristics can be adopted in the search for feasible points before the algorithm is executed.

Table 1
Computational results from CEM.

Pr.	n	$f(x^*)$	δ	β	Best	Average	Worst
G1	13	−15.000	0.01	0.6	−15.000	−15.000	−15.000
G2	20	−0.803619	0.01	0.6	−0.623711	−0.517221	−0.4522348
G3	5	−1.0	0.01	0.6	−1.00151	−1.00167	−1.00176
G4	5	−30 665.539	0.1	0.6	−30 665.513	−30 660.649	−30 654.500
G5	4	5126.4981	0.01	0.8	5126.4842	5128.6958	5136.6618
G6	2	−6961.81388	0.001	0.5	−6961.813	−6961.813	−6961.813
G7	10	24.3062091	0.1	0.5	25.11276	27.75496	29.93511
G8	2	−0.095825	0.01	0.5	−0.095825	−0.095825	−0.095825
G9	7	680.630057	0.01	0.4	680.8968	681.3511	681.7680
G10	8	7049.3307	0.1	0.7	7049.7581	7154.6709	7292.7241
G11	2	0.75	0.001	0.5	0.7499	0.7499	0.7499
G12	3	−1.0	0.01	0.5	−1.0000	−1.0000	−1.0000
G13	5	0.0539498	0.01	0.5	0.053827	0.056314	0.059852

Table 2
Comparison between CEM and other constrained EM.

Pr.	$f(x^*)$		PEM	FADEM	CEM
		Best	−14.957990	−14.99980	−15.0000
G1	−15.0	Average	−12.624385	−14.59047	−15.0000
		Worst	−11.028430	−12.50051	−15.0000
		Best	−0.516353	−0.444939	−0.623711
G2	−0.803619	Average	−0.453309	−0.427705	−0.517221
		Worst	−0.404182	−0.377406	−0.452234
		Best	−1.00215	−1.00298	−1.00151
G3	−1.0	Average	−1.00008	−0.99966	−1.00167
		Worst	−0.99553	−0.99643	−1.00176
		Best	−30 661.790	−30 642.600	−30 665.513
G4	−30 665.539	Average	−30 642.891	−30 591.897	−30 660.649
		Worst	−30 627.758	−30 521.434	−30 654.500
		Best	5126.4218	5135.818	5126.4842
G5	5126.4981	Average	5191.5619	5338.583	5128.6958
		Worst	5384.7132	6117.660	5136.6618
		Best	−6961.77	−6953.41	−6961.813
G6	−6961.81388	Average	−6961.74	−6942.91	−6961.813
		Worst	−6961.71	−6933.26	−6961.813
		Best	26.14955	27.46692	25.11276
G7	24.3062091	Average	28.76543	53.08965	27.75496
		Worst	32.74971	103.5507	29.93511
		Best	−0.095825	−0.095825	−0.095825
G8	−0.095825	Average	−0.095825	−0.095825	−0.095825
		Worst	−0.095825	−0.095825	−0.095825
		Best	681.2211	681.777	680.8968
G9	680.630057	Average	682.6084	689.896	681.3511
		Worst	685.1694	701.344	681.7680
		Best	7100.8486	7187.18	7049.7581
G10	7049.3307	Average	7226.7802	9492.73	7154.6709
		Worst	7434.9652	16395.51	7292.7241
		Best	0.7490	0.749001	0.7499
G11	0.75	Average	0.7490	0.749079	0.7499
		Worst	0.7490	0.749346	0.7499
		Best	−1.0000	−1.0000	−1.0000
G12	−1.0	Average	−1.0000	−1.0000	−1.0000
		Worst	−1.0000	−1.0000	−1.0000
		Best	0.346621	0.063196	0.053827
G13	0.0539498	Average	0.911530	2.083421	0.056314
		Worst	2.124983	20.784428	0.059852

parameters need to be defined. These are the size N of the population set S_k, the local search parameters δ and I_ℓ, and the parameter β used in (14). We have used $N = 10$ and $I_\ell = 10$. With these values we test CEM to determine some suitable parameter values for δ and β. Our numerical experiments have shown that δ is much more sensitive (with respective to the quality of optimal solution) than β. CEM performs well with a wide range of β values (given a suitable choice of δ), but we have only presented our results in the next table for a combination of δ and β for each problem. Results presented in this section are based on 30 independent runs, and each run of CEM is terminated after 350 000 function calls. Throughout the numerical experiments, we convert all equality constraints, $h(x) = 0$, into inequality constraints by using

$$|h(x)| - \varepsilon \leq 0 \tag{16}$$

Table 3
Comparison between CEM and recent PSO methods.

Pr.	$f(x^*)$		SAVPSO	CHMPSO	PESO	PSO	CEM
		Best	−15.0000	−15	−15.0000	−15.0000	−15.0000
G1	−15.0	Average	−14.7151	−15	−15.0000	−14.7104	−15.0000
		Worst	−12.4531	−15	−15.0000	−13.0000	−15.0000
		Best	−0.803443	−0.803432	−0.792608	−0.669158	−0.623711
G2	−0.803619	Average	−0.740577	−0.790406	−0.721749	−0.419960	−0.517221
		Worst	−0.631598	−0.750393	−0.614135	−0.299426	−0.452234
		Best	1.0048	1.00472	1.00501	0.99393	−1.00151
G3	−1.0	Average	1.0034	1.00381	1.00501	0.76481	−1.00167
		Worst	0.9976	1.00249	1.00499	0.46401	−1.00176
		Best	−30 665.539	−30 665.500	−30 665.539	−30 665.539	−30 665.513
G4	−30 665.539	Average	−30 665.539	−30 665.500	−30 665.539	−30 665.539	−30 660.649
		Worst	−30 665.539	−30 665.500	−30 665.539	−30 665.539	−30 654.500
		Best	5126.4842	5126.6400	5126.484	5126.484	5126.4842
G5	5126.4981	Average	5202.3627	5461.0813	5129.178	5135.973	5128.6958
		Worst	5520.1467	6104.7500	5148.859	5249.825	5136.6618
		Best	−6961.81	−6961.81	−6961.81	−6961.81	−6961.813
G6	−6961.81388	Average	−6961.81	−6961.81	−6961.81	−6961.81	−6961.813
		Worst	−6961.81	−6961.81	−6961.81	−6961.81	−6961.813
		Best	24.319	24.35110	24.30692	25.11276	25.11276
G7	24.3062091	Average	24.989	25.35577	24.37125	27.75496	27.75496
		Worst	26.194	27.31680	24.59350	29.93511	29.93511
		Best	−0.095825	−0.095825	−0.095825	−0.095825	−0.095825
G8	−0.095825	Average	−0.095825	−0.095825	−0.095825	−0.095825	−0.095825
		Worst	−0.095825	−0.095825	−0.095825	−0.095825	−0.095825
		Best	680.632	680.638	680.630	680.630	680.8968
G9	680.630057	Average	680.653	680.852	680.630	680.630	681.3511
		Worst	680.699	681.553	680.630	680.631	681.7680
		Best	7054.1256	7057.5900	7049.4594	7049.3809	7049.7581
G10	7049.3307	Average	7173.2661	7560.0478	7099.1014	7205.5000	7154.6709
		Worst	7335.2477	8104.3100	7251.3962	7894.8124	7292.7241
		Best	0.749	0.749999	0.7490	0.7490	0.7499
G11	0.75	Average	0.749	0.750107	0.7490	0.7490	0.7499
		Worst	0.749	0.752885	0.7490	0.7490	0.7499
		Best	−1.0	−1.0000	−1.0000	−1.0000	−1.0000
G12	−1.0	Average	−1.0	−1.0000	−1.0000	−0.9989	−1.0000
		Worst	−1.0	−1.0000	−1.0000	−0.9944	−1.0000
		Best	0.053866	0.068665	0.081498	0.085655	0.053827
G13	0.0539498	Average	0.552753	1.716426	0.626881	0.569358	0.056314
		Worst	1.856102	13.66950	0.997586	1.793161	0.059852

with $\varepsilon = 10^{-3}$. The first set of results obtained from CEM is presented in Table 1, which shows that the optimal solutions are obtained with reasonably good accuracies although we have used a simple coordinate direction local search suggested by Birbil and Fang [5] for unconstrained optimization. In Table 1 and in all subsequent tables, a value better than the best known optimal value for G3 is presented. This has occurred due to the value of ε used in (16).

4.1. Comparison of CEM with other constrained EM

We now compare CEM with two of its competitors. They are the penalty-based EM (PEM) [11] and the feasibility and domain rule-based EM (FDEM) [10]. For the purpose of a fair comparison, we have coded PEM and run it for the same number of function calls, i.e. 350 000 function calls. Results from CEM and PEM are presented in Table 2, where we have also included the summarized results from FDEM, using 30 independent runs. The results from FDEM are taken from [10]; these results are also obtained for 350 000 function calls of FDEM. A comparison using the best optimal values found for CEM and PEM shows that CEM is superior to PEM for all problems except for G3, G8 and G12 where the two algorithms are equally matched. Results also show that CEM outperforms FDEM. The superiority of CEM over PEM and FDEM holds.

4.2. Comparison of CEM with PSO

Here, we compare CEM with four different particle swarm optimization (PSO) algorithms designed for constrained optimization. The results for these algorithms (PSO, PESO [12], CHMPSO [13] and SAVPSO [14]) are taken from the corresponding references. PSO, PESO and CHMPSO use 350 000 function calls as the termination criterion. CEM also uses 350 000 function calls. On the other hand, SAVPSO uses 50 000 function calls as its termination criterion. Nonetheless, we have included this in our comparison. The summarized results for all algorithms using 30 independent runs are presented in Table 3. Table 3 shows that CEM produces slightly better results compared to the rest of the algorithms for G1, G6, G10 and G13. Algorithms

are competitive on the remaining problems. Although SAVPSO uses fewer function calls it has a preprocessing phase for treating constraint violations. It is therefore clear that CEM has a role to play in constrained global optimization.

5. Conclusions and further research

In this study we proposed a constrained version of the electromagnetism-like method and compared its performance with those of two previously developed electromagnetism-like methods and four other methods based on particle swarm optimization. The comparisons presented established the superiority of the new algorithm over other two electromagnetism-like methods. The new algorithm is also comparable to all particle swarm optimization methods.

A weakness of the new electromagnetism-like method proposed here is the local search that we have implemented within the method. We feel that the performance of the new algorithm can be improved significantly with a number of changes including the incorporation of an efficient local search.

Further research in under way for developing a more efficient electromagnetism-like method for addressing constrained problems.

References

[1] K. Miettinen, M. Makela, J. Toivanen, Numerical comparison of some penalty based constraint handling techniques in genetic algorithm, Journal of Global Optimization 27 (2003) 427–446.
[2] T.P. Runarsson, X. Yao, Stochastic ranking for constrained evolutionary optimization, IEEE Transactions on Evolutionary Computation 4 (3) (2000) 284–294.
[3] D. Powell, M.M. Skolnick, Using genetic algorithms in engineering design optimization with non-linear constraints, in: Proceedings of the Fifth International Conference on Genetic Algorithms, 1993, pp. 424–430.
[4] K. Deb, An efficient constraint handling method for genetic algorithms, Computer Methods in Applied Mechanics and Engineering 186 (2000) 311–338.
[5] S.I. Birbil, S.C. Fang, An electromagnetism-like mechanism for global optimization, Journal of Global Optimization 25 (3) (2002) 263–282.
[6] E.W. Cowan, Basic Electromagnetism, Academic Press, New York, 1968.
[7] S.I. Birbil, S.C. Fang, R.L. Sheu, On the convergence of a population-based global optimization algorithm, Journal of Global Optimization 30 (2005) 301–318.
[8] M. Golalikhani, N. Javadian, R. Tavakkoli-Moghaddam, A novel hybrid approach combining electromagnetism-like method with Solis and Wets local search for continuous optimization problems, Journal of Global Optimization 44 (2009) 227–234.
[9] A.M.A.C. Rocha, E.M.G.P. Fernandes, Modified movement force vector in a electromagnetism-like mechanism for global optimization, Optimization Methods and Software 24 (2009) 253–270.
[10] A.M.A.C. Rocha, E.M.G.P. Fernandes, Feasibility and dominance rules in the electromagnetism-like algorithm for constrained global optimization, Lecture Notes in Computer Science 5071 (2008) 768–783.
[11] S.I. Birbil, Stochastic global optimization techniques, Ph.D. Thesis, North Carolina State University, 2002.
[12] A.E.M. Zavala, A.H. Aguirre, E.R.V. Diharce, Constrained optimization via particle evolutionary swarm optimization algorithm (PESO), GECCO (2005) 209–216.
[13] G. Toscano, C.A. Coello Coello, A constraint-handling mechanism for particle swarm optimization, in: Proceedings of the 2004 IEEE Congress on Evolutionary Computation, 2004, pp. 1396–1403.
[14] H. Lu, W. Chen, Self adaptive velocity particle swarm optimization for constrained optimization problems, Journal of Global Optimization 41 (2008) 427–445.